

Amendments to the Claims:

This claims listing replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Original) A method for optimizing dependencies for a set of objects comprising:
automatically detecting dependencies among a set of objects, wherein each of said objects in said set includes at least one linkable file;
adding said detected dependencies to a dependency list for said set of objects; and
removing dependencies from said dependency list for any object that does not also have at least one file dependency.
2. (Original) A method for optimizing dependencies as recited in claim 1 further comprising removing unused files from said set of objects.
3. (Original) A method for optimizing dependencies for a set of objects as recited in claim 2 further comprising breaking a selected object in said set of objects into at least two smaller objects if said selected object is greater than a maximum object size.
4. (Original) A method for optimizing dependencies for a set of objects as recited in claim 3 wherein said threshold maximum size is a predetermined maximum object size.
5. (Original) A method for optimizing dependencies for a set of objects as recited in claim 3 further comprising making a selected file into a new object if the number of dependencies of said selected file is greater than a maximum file dependency number.
6. (Original) A method for optimizing dependencies for a set of objects as recited in claim 5 wherein said maximum file dependency number is a predetermined maximum file dependency number.
7. (Original) A method for optimizing dependencies for a set of objects as recited in claim 6 further comprising manually editing said dependency list.
8. (Original) A method for optimizing dependencies for a set of objects as recited in claim 1 further comprising manually detecting dependencies among said set of objects, and adding said manually detected dependencies to said dependency list.

9. (Original) A method for optimizing dependencies for a set of objects as recited in claim 1 wherein automatically detecting dependencies among a set of objects comprises:

recording dependencies to create a list of recorded dependencies during a traversal of said set of objects; and

analyzing said list of recorded dependencies to automatically detect dependencies.

10. (Original) A method for optimizing dependencies for a set of objects as recited in claim 1 further comprising manually editing said dependency list.

11. (Original) An apparatus for optimizing dependencies for a set of objects comprising:

means for automatically detecting dependencies among a set of objects, wherein each of said objects in said set includes at least one linkable file;

means for adding said detected dependencies to a dependency list for said related objects; and

means for removing dependencies from said dependency list for any object that does not also have at least one file dependency.

12. (Original) An apparatus for optimizing dependencies as recited in claim 11 further comprising means for removing unused files from said set of objects.

13. (Original) An apparatus for optimizing dependencies for a set of objects as recited in claim 12 further comprising means for breaking a selected object in said set of objects into at least two smaller objects if said selected object is greater than a maximum object size.

14. (Original) An apparatus for optimizing dependencies for a set of objects as recited in claim 13 wherein said threshold maximum size is a predetermined maximum object size.

15. (Original) An apparatus for optimizing dependencies for a set of objects as recited in claim 13 further comprising means for making a selected file into a new object if the number of dependencies of said selected file is greater than a maximum file dependency number.

16. (Original) An apparatus for optimizing dependencies for a set of objects as recited in claim 15 wherein said maximum file dependency number is a predetermined maximum file dependency number.

17. (Original) An apparatus for optimizing dependencies for a set of objects as recited in claim 16 further comprising means for manually editing said dependency list.

18. (Original) An apparatus for optimizing dependencies for a set of objects as recited in claim 11 further comprising means for manually detecting dependencies among said set of objects, and means for adding said manually detected dependencies to said dependency list.

19. (Original) An apparatus for optimizing dependencies for a set of objects as recited in claim 11 wherein said means for automatically detecting dependencies among a set of objects comprises:

means for recording dependencies to create a list of recorded dependencies during a traversal of said set of objects; and

means for analyzing said list of recorded dependencies to automatically detect dependencies.

20. (Currently Amended) An apparatus for optimizing dependencies for a set of objects as recited in claim [[1]] 11 further comprising means for manually editing said dependency list.

21. (Withdrawn) A method for providing a tutorial comprising:

developing a course from an initial set of objects each including at least one linkable file, said initial set of objects being improved for at least one of transmission and storage purposes by the automatic detection of dependency information with regards to said initial set of objects and the use of said dependency information to at least one of modify an object, remove an object, split an object, and form an object to develop an improved set of objects comprising said course; and

playing said course for a student.

22. (Withdrawn) A method for providing a tutorial as recited in claim 21 further comprising receiving a request for said course by said student.

23. (Withdrawn) A method for providing a tutorial as recited in claim 22 further comprising developing a course list for presentation to said student.

24. (Withdrawn) A method for providing a tutorial as recited in claim 23 further comprising storing said course and said course list in a publishing database.

25. (Withdrawn) A method for providing a tutorial as recited in claim 21 further comprising storing multiple versions of said course in a master repository.

26. (Withdrawn) Computer readable media including code segments for providing a tutorial comprising:

a code segment for developing a course from an initial set of objects each including at least one linkable file, said initial set of objects being improved for at least one of transmission and storage purposes by the automatic detection of dependency information with regards to said initial set of objects and the use of said dependency information to at least one of modify an object, remove an object, split an object, and form an object to develop an improved set of objects comprising said course; and

a code segment for playing said course for a student.

27. (Withdrawn) Computer readable media including code segments for providing a tutorial as recited in claim 26 further comprising a code segment for receiving a request for said course by said student.

28. (Withdrawn) Computer readable media including code segments for providing a tutorial as recited in claim 27 further comprising a code segment for developing a course list for presentation to said student.

29. (Withdrawn) Computer readable media including code segments for providing a tutorial as recited in claim 28 further comprising a code segment for storing said course and said course list in a publishing database.

30. (Withdrawn) Computer readable media including a code, segment for providing a tutorial as recited in claim 26 further comprising storing multiple versions of said course in a master repository.

31. (Withdrawn) An authoring environment comprising:
a repository storing a set of objects comprising a course, each of said objects
including at least one file;
a content player coupled to said repository for receiving and playing said course, said
content player including a dependency recorder which develops a dependency list with
regards to said set of objects as they are played;
an editor coupled to said content player and capable of editing said set of objects, said
editor including a dependency analyzer which uses, at least in part, said dependency list to
provide a dependency analysis for said course.

32. (Withdrawn) An authoring environment as recited in claim 31 further comprising
a repository explorer coupling said content player to said repository.

33. (Withdrawn) An authoring environment as recited in claim 32 wherein said
repository explorer includes a dependency editor which modify can an object, remove an
object, split an object, and form an object to develop an improved set of objects for said
course.